TRIP: A Low-Cost Vision-Based Location System for Ubiquitous Computing

Diego López de Ipiña[1], Paulo R. S. Mendonça[2] and Andy Hopper[1,3]

[1]Laboratory for Communications Engineering, University of Cambridge, UK; [2]Fallside Laboratory, University of Cambridge, UK; [3]AT&T Laboratories Cambridge, UK

Abstract: Sentient Computing provides computers with perception so that they can react and provide assistance to user activities. Physical spaces are made sentient when they are wired with networks of sensors capturing context data, which is communicated to computing devices spread through the environment. These devices interpret the information provided and react by performing the actions expected by the user. Among the types of context information provided by sensors, *location* has proven to be especially useful. Since location is an important context that changes whenever the user moves, a reliable location-tracking system is critical to many sentient applications. However, the sensor technologies used in indoor location tracking are expensive and complex to deploy, configure and maintain. These factors have prevented a wider adoption of Sentient Computing in our living and working spaces. This paper presents TRIP, a low-cost and easily deployable vision-based sensor technology addressing these issues. TRIP employs off-the-shelf hardware (low-cost CCD cameras and PCs) and printable 2-D circular markers for entity identification and location. The usability of TRIP is illustrated through the implementation of several sentient applications.

Keywords: Computer vision; Context-aware computing; CORBA; Location-aware computing; Sentient spaces

1. Introduction

Ubiquitous Computing [1] envisions physical spaces, such as offices or homes that are augmented with computing devices integrated into the environment. It aims to make services provided by these devices as commonly available to end-users as electricity. It corresponds to the next generation of computing in which the user's attention will not be drawn to computers but *vice versa*, i.e. computers will be attentive to user interactions and aid their daily activities.

Sentient Computing [2, 3] is our approach to making Ubiquitous Computing a reality. It creates perceptive living spaces [4] where users activities are enhanced by software services provided by devices embedded in the environment. These environments achieve awareness of their surroundings through sensors that capture contextual information such as the location and identity of objects or the ambient sound level and temperature of a physical space. Sentient Computing combines the dynamic information conveyed by sensors with static information from data repositories (e.g. entity attributes, the geometric features of a physical location or the capabilities of devices) in order to build a model of the environment's state. Assisted by such model, sentient systems aim to perform the right service at the right time on behalf of users. In essence, computers are given perception so that they can become reactive to the people and activities taking place around them.

Location-aware Computing [5], whose behaviour is determined by the position of objects in the environment, represents an interesting subset of the Sentient Computing paradigm. The location attribute of context can be used to convert the space users inhabit into a global interface with computers [6], rather than limiting users to the classical desktop-bound interaction. In the last few years, several custom-built indoor location systems have appeared providing different entity location granularity. The location information provided ranges from room-scale resolution such as the infrared-based Active Badge [5], to more accurate 3-D coordinate resolution offered by systems such as the ultrasound-based Bat [7] or the radio-based 3-D-iD [8]. Most of these indoor location systems require people and objects to be located to wear an electronic tag that transmits a unique identifier via either an infrared, ultrasound or radio interface to a network of sensors on the walls or ceilings of a building. A *location server* polls and analyses the information from the sensors and makes it available to applications.

Despite their proved usability, most of existing indoor location systems [9] have some important drawbacks. The tags they use are proprietary and need battery power to operate. The infrastructure required, i.e. a network of sensors, is complex and expensive to install and maintain. The cost of these networks of sensors is usually around $1000–2000 per room. These factors have limited the deployment of such positioning systems to research laboratories. We have devised an alternative sensor technology, known as TRIP that is deemed to provide a better trade-off between the price and flexibility and the accuracy of the location data provided. In what follows, a detailed overview is given of the TRIP technology and its main areas of application.

2. A Vision-Based Location Sensor

TRIP (Target Recognition using Image Processing) is a vision-based sensor system that uses a combination of 2D circular barcode tags or *ringcodes* (see Fig. 1), and inexpensive CCD cameras (e.g. web-cams or CCTV cameras) to identify and locate tagged objects in the cameras' field of view. Compared with untagged vision-based location systems [10, 11], the processing demands of TRIP are low. Optimised image processing and computer vision algorithms are applied to obtain, in real-time, the identifier (*TRIPcode*) and *pose* (location and orientation) of a target with respect to the viewing camera.

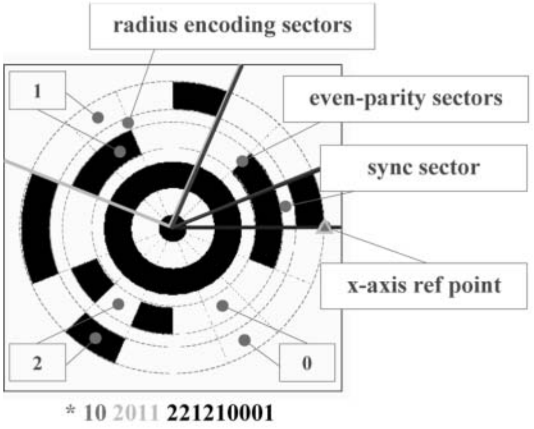


Fig. 1. TRIPcode of radius 58mm and ID 18,795.

TRIP is a cost-efficient and versatile sensor technology. Only low-cost (e.g. web-cams below $50), low-resolution (640 × 480 pixel) cameras are required. The spare computing resources within a LAN can be allocated to run the TRIP image parsing software, without the need of investing in new PCs. Commonly available monochromatic printers can be used to print out the black and white TRIPtags. The off-the-shelf hardware requirements and software nature of TRIP make possible its installation in offices, or event at homes, equipped with some PCs, without requiring significant additional costs or expert technical knowledge. A medium size (4 × 4 m^2) room could be visually covered with three or four cameras.

TRIP's 2-D printable and resizable ringcodes (*TRIPtags*) encode a ternary number in the range 1–19,683($3^9 - 1$) in the two concentric rings surrounding the bull's-eye of a target. These two rings are divided into 16 sectors. The first sector or synchronisation sector indicates where the TRIPcode begins. This sector presents black areas in its section of the encoding rings, a configuration impossible everywhere else in the code. The code is read in anti-clockwise fashion from this sector. The subsequent two sectors are reserved for undertaking an even parity check on the identifier (TRIPcode) extracted. The following four sectors encode the radius of its central bull's-eye in millimetres. Finally, the remaining nine sectors encode a ternary identifier.

The low-cost of TRIPtags and their ample addressing space (19,683 possible codes) make them suitable for tagging even low-cost items, such as books or office stationary. In contrast, other location sensing technologies can only tag more valuable assets, *e.g.* personnel or equipment, given the high cost (usually $10–50) and thickness (1–2 cm) of the associated tags. Notably, the main constraint of TRIP is its requirement for a direct line of sight between the camera and the tag. However, we believe the advantageous features of this technology over other alternative location technologies offset this important drawback. The ultimate goal of this project is to provide a downloadable version of the system so that users with web-cams attached to their PCs may install the TRIP software and hence provide visual awareness to their machines.

2.1. Target Design Criteria

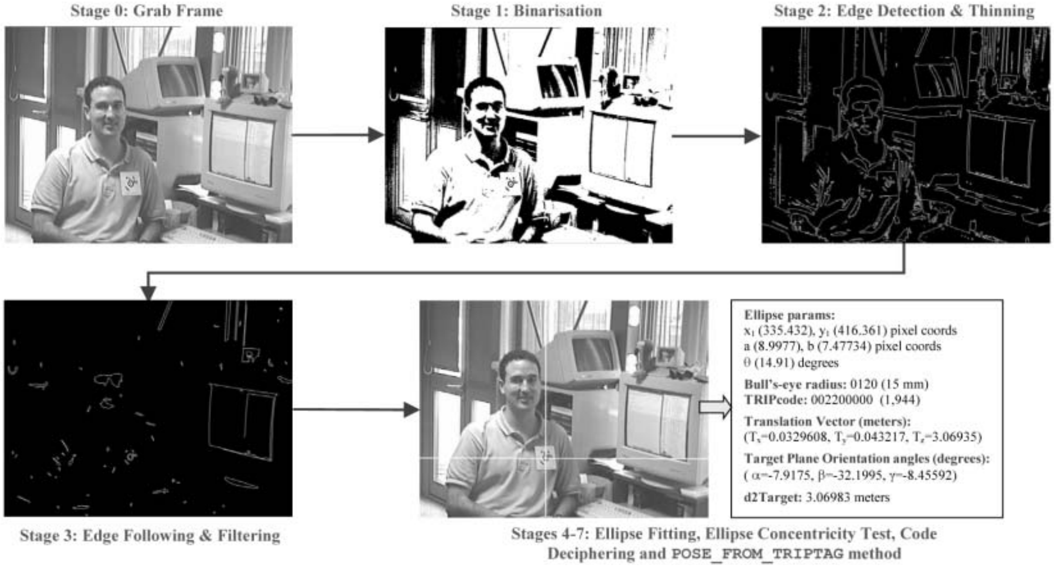
Conventionally, researchers have used visual cues (markers) of known geometric properties to facilitate the identification and localization of objects in the field of view of video cameras. Markers of known size and shape can yield a wealth of geometric information when transformed under the perspective projection. In particular, circular [12] and square [13, 14] markers have been utilised to extract the accurate position and orientation of tagged objects. These markers permit the application of simple and fast-performing image processing and computer vision algorithms to the difficult task of obtaining 3-D position from a 2D image.

Our marker (see Fig. 1) is based on a circular pattern since circles are less ambiguous than squares for their identification and 3D localization using a visual sensor. In man-made environments, circles are less common shapes than right angles, squares and rectangles. TRIP is intended for the localization of tagged objects in these highly cluttered environments (see Fig. 2), where the central bull's-eye of a TRIPtag represents a very distinctive pattern. The detection of squares in those cluttered environment supposes an expensive computational task given the many straight edges combinations possible within an image. An even more unusual shape could have chosen, e.g. a star. In this case, however, there is the problem of representing such a shape in a compact way and to encode useful information (e.g. ID) around or within it. A circle, and its projection in an image as an ellipse, can be represented in a compact and elegant form by a symmetric (3 × 3) matrix. Moreover, ellipses in an image are salient features and can be detected robustly and rapidly by methods such as those of Pilu et al. [15].

A second important aspect in the design of a marker, apart from its shape, is the set of colours used in order to provide a distinctive contrast between the marker and the surrounding objects. TRIP targets were chosen to be black and white patterns since: (1) it is computationally cheaper to process monochromatic images rather than colour ones, and (2) monochromatic printers are also more widely available than colour ones. Using colour is both difficult and unreliable since the lighting conditions can change too much and the colour sensitivity of various CCD devices are very different.

2.2. TRIP Target Recognition

TRIP applies a target recognition algorithm to the raw video data supplied by a camera in order to determine the identifiers and geometric properties of the projections of TRIPtags in a frame. This procedure converts the TRIP system into an entity identification sensor. Figure 2 depicts the video filtering process undertaken to an image containing a TRIPtag. The set of image


Stage 0: Grab Frame
Stage 1: Binarisation
Stage 2: Edge Detection & Thinning
Stage 3: Edge Following & Filtering
Stages 4-7: Ellipse Fitting, Ellipse Concentricity Test, Code Deciphering and POSE_FROM_TRIPTAG method

```
Ellipse params:
x1 (335.432), y1 (416.361) pixel coords
a (8.9977), b (7.47734) pixel coords
θ (14.91) degrees
Bull's-eye radius: 0120 (15 mm)
TRIPcode: 002200000  (1,944)
Translation Vector (meters):
(Tx=0.0329608, Ty=0.043217, Tz=3.06935)
Target Plane Orientation angles (degrees):
( α=-7.9175, β=-32.1995, γ=-8.45592)
d2Target: 3.06983 meters
```

Fig. 2. Identifying and locating the TRIPtag attached to a user's shirt in a 768x576 pixel image.

processing stages applied to each greyscale image grabbed are:

1. The original image is transformed into a binary image with increased contrast level by applying an *adaptive thresholding* procedure [16]. This stage makes TRIP's image analysis very robust to variable lighting conditions and suitable even for low-cost, low-quality CCTV or web-cams.
2. One-pixel width edges in the binary image are extracted by applying a very computationally efficient binary edge detection method. This method applies the following criteria to a binary image: "*A pixel is an edge point if it has black intensity value and an adjacent (4-connected) neighbour pixel with white intensity value and a diagonally adjacent (8-connected) one with black intensity value*".
3. All the 8-connected chains of edge points (*edgel*) are followed in a clockwise fashion, producing for each edge a list of ordered point locations. TRIP is only concerned with edges corresponding to the perspective projections of the circular borders of TRIPtags, i.e. ellipses. Thus, only edges whose shape is likely to define an elliptical shape are kept. For this, a simple heuristic is applied that analyses the ratio between the Euclidean distance of the extremes of an edge and its length in pixels.
4. The direct least-squares ellipse fitting method developed by Pilu et al. [15] is applied to each elliptical edge candidate encountered, obtaining the ellipse parameters that best approximate the edge points.
5. The ellipses found are tested for concentricity in order to identify candidate TRIPcodes.
6. The code deciphering stage is undertaken. This method applies an efficient pixel-sampling mechanism on the binary image result of stage 1, based on the parameters of the outermost ellipse of a candidate TRIPcode. The ternary codes obtained are validated against the two even parity check bits reserved in a TRIPcode.
7. For each TRIPtag spotted within an image TRIP returns: (a) the ternary identifier, (b) the parameters of the implicit equation of the outermost elliptical border of the projection of target, (c) the radius of the target and (d) the image location of the bottom outermost corner of the synchronisation sector of a TRIPtag (this point is identified during stage 6).

2.3. TRIP Target Pose Extraction

The TRIP sensor system applies a model-based object location technique [17] to a single intensity image in order to extract the pose of TRIP-tagged objects with respect to a viewing camera. The 'TRIPtag object model' (see Fig. 3) is composed of: (1) the outermost circular border of the central bull's-eye and (2) a point outside the bull's-eye (denoted by P in Fig. 3), lying on the X-axis of the target centred coordinate system. This point corresponds to the bottom outermost corner of the synchronisation sector of a TRIPtag.

The model-based object location technique used, namely `POSE_FROM_TRIPTAG`, takes as input the outcome of the target recognition process in order to determine the accurate 3D location and orientation of identified TRIPtags with respect to the viewing cameras. This method establishes a transformation that back-projects the elliptical projection of the outermost border of a target into its actual circular form, of known radius, in the target plane. The radius of the outermost border of a target's bull's-eye is encoded within the four ternary bits reserved for this purpose in a TRIPcode. In addition, the image location of the bottom outermost corner of the synchronisation sector of a TRIPtag is necessary. This point is situated on the X-axis of the target centred coordinate system.

The `POSE_FROM_TRIPTAG` method returns the translation vector $[T_x\ T_y\ T_z]$ and rotation angles (α, β, γ) that define the rigid body transformation between a camera coordinate system and a target centred coordinate system. The bottom right-hand side of Fig. 2 shows the

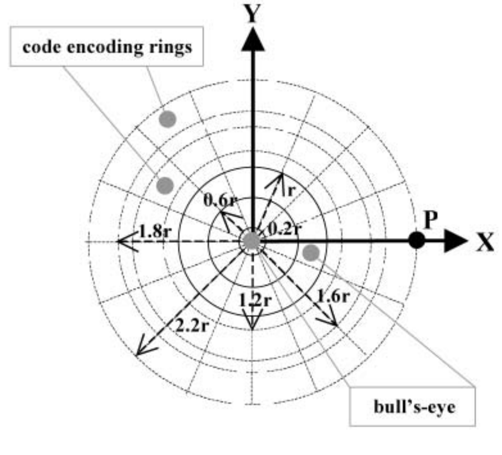


Fig. 3. TRIPtag geometric model.

results of applying such process to an image. This method exploits the property unique to a conic that is a circle that the back-projected curve's implicit equation must have equal coefficients x^2 and y^2 and no term in xy. The algorithm was inspired by the `POSE_FROM_CIRCLE` method described in Forsyth et al. [17].

The geometry applied by the `POSE_FROM_TRIPTAG` method is shown in Figs 4 and 5. The ellipse corresponding to the image of the outermost circular border of the bull's-eye of a target defines a cone with its vertex at the centre of projection of the pinhole camera (O). The orientation of the target's plane, Π_t, is found by rotating the camera so that the intersection of the cone with the image plane becomes a circle, which happens when the image plane, Π_i, is parallel to the target plane, Π_t. This rotation can be estimated as the composition of two successive rotations. The first (see Fig. 4) is a 3D rotation that puts the Z axis through the centre of the target, and aligns the X and Y axes with the axes of the image ellipse; the second (see Fig. 4) rotates the newly found coordinate system around Y' until the image plane becomes parallel to Π_t.

TRIP makes the assumption that the intrinsic parameters of the camera are known. In other words, the user is required to undertake the well-known camera calibration [19] process once for each CCD camera. Fortunately, there are several publicly available software packages for camera calibration [19, 20]. The following sub-sections explain thoroughly the projective geometry used to find the pose of a TRIPtag in the scene.

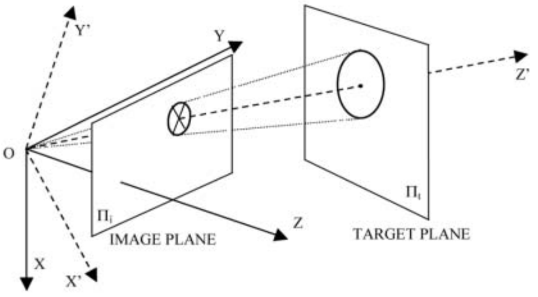


Fig. 4. 3D rotation of (X,Y,Z) to the eigenvector frame (X', Y', Z').

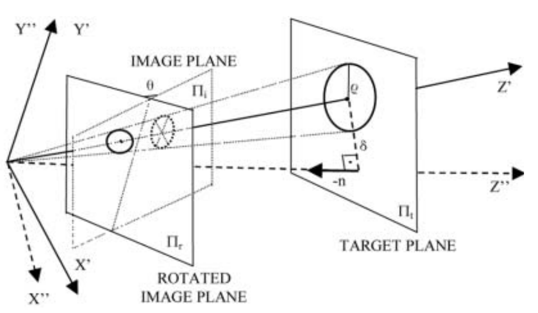


Fig. 5. Rotation of plane Π_i about the Y' axis by an angle θ.

2.3.1. Target Plane Orientation

The `POSE_FROM_TRIPTAG` method assumes that the ellipse in the image taken as reference is expressed in the camera coordinate system rather than in the image plane pixel coordinates as returned by the target recognition process. Consequently, the ellipse, C, is normalised by using the matrix, K, of the camera intrinsic parameters (see Eq. (1)). The intrinsic parameters of a camera are defined as the focal length, f, the intersection of the optical axis (Z-axis in Fig. 4) with the image plane (u_0, v_0), the effective pixel size in the horizontal and vertical direction (k_u, k_v), and the pixel skew k_α. For most CCD cameras the pixels are almost perfectly rectangular and thus $k_\alpha \approx 0$. Note that the superscript T in Eq (1) stands for transpose:

$$K = \begin{bmatrix} k_u \cdot f & k_\alpha \cdot f & u_0 \\ 0 & k_v \cdot f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \quad C_n = K^T \cdot C \cdot K \qquad (1)$$

The normalized ellipse, C_n, corresponding to the projection of the outermost circular border of a target's bull's-eye, defines a cone with vertex in the centre of projection of the pinhole camera, denoted by O (see Fig. 4). Let $ax^2+bxy+cy^2+dx+ey+f = 0$ be the equation of this normalized ellipse, then there is a cone in 3D defined as:

$$ax^2+bxy+cy^2+dxz+eyz+fz^2 = P^T \cdot C_n \cdot P = 0 \qquad (2)$$

where $P = [X\ Y\ Z]^T$ is a point in the cone and C_n is the real symmetric matrix of a cone:

$$C_n = \begin{bmatrix} a & b/2 & d/2 \\ b/2 & c & e/2 \\ d/2 & e/2 & f \end{bmatrix} \qquad (3)$$

The orientation of the circle's plane, Π_t, is found by rotating the camera so that the intersection of the cone with the image plane becomes a circle, which happens when the image plane, Π_i, is parallel to the target plane. This rotation is estimated as the composition of two successive rotations, namely R_1 and R_2:

$$R_C = R_1 \cdot R_2 \tag{4}$$

The first rotation, R_1, is determined by diagonalising C_n, *i.e.* removing the coefficients with terms in *xy*, *xz* and *yz*. This 3D rotation (see Fig. 4) transforms the camera coordinate system, *OXYZ*, to the eigenvector frame, OX'Y'Z', and the ellipse matrix C_n into C'. If λ_1, λ_2, λ_3 are the eigenvalues of C_n, with $\lambda_1 < \lambda_2 < \lambda_3$ and $\vec{e}_1$, $\vec{e}_2$, $\vec{e}_3$, the corresponding eigenvectors, then:

$$\vec{P'} = R_1^{\,T} \cdot \vec{P} \qquad R_1 = \left[\vec{e_1} \quad \vec{e_2} \quad \vec{e_3}\right]$$

$$C' = R_1^T \cdot C_n \cdot R_1 = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \tag{5}$$

The second rotation, R_2, is obtained by imposing the equality of the coefficients of x^2 and y^2 in C'. R_2 represents a rotation around the Y' axis by an angle θ, as shown in Eq. (6). This rotation, R_2, sends a point *P*' to *P*'' and transforms C' into C'' (see Fig. 5):

$$\theta = \pm\tan^{-1}\sqrt{\frac{\lambda_2 - \lambda_1}{\lambda_3 - \lambda_2}}$$

$$R_2 = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{6}$$

$$C'' = R_2^T \cdot C' \cdot R_2 \qquad \vec{P''} = R_2^{\,T} \cdot \vec{P'} = R_2^T \cdot R_1^T \cdot \vec{P}$$

The composite rotation R_C, result of multiplying R_1 and R_2, makes the image plane parallel to the target plane (see Fig. 5). Consequently a vector normal to the target plane can be obtained by applying Eq. (7). Vector $\vec{n}$ represents the orientation of the plane Π_t expressed in the camera coordinate system.

$$\vec{n} = R_C \cdot \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T = \begin{bmatrix} R_{13} & R_{23} & R_{33} \end{bmatrix}^T \tag{7}$$

As result of Eq. (6), there is a two-fold ambiguity in the recovered orientation, depending on which sign of θ is chosen, *i.e.* $\vec{n}_1$ from R_{C1} and $\vec{n}_2$ from R_{C2}. Section 2.3.3. explains the geometric interpretation of this result and a procedure to break this ambiguity.

2.3.2. Target 3-D Location

C'' represents a circle in the rotated image plane (Π_r) of radius r_0 and centre (x_0, *0*, *1*), in terms of the coordinate system OX''Y''Z'' (see Fig. 6):

$$C'' = \begin{bmatrix} 1 & 0 & -x_0 \\ 0 & 1 & 0 \\ -x_0 & 0 & x_0^{\,2} - r_0^{\,2} \end{bmatrix} \tag{8}$$

If correspondences are established between the expressions for C'' is Eqs (6) and (8), then the following expressions can be obtained to calculate x_0 and r_0:

$$x_0^{\,2} = \frac{(\lambda_2 - \lambda_1)(\lambda_3 - \lambda_2)}{\lambda_2^2} \qquad r_0^{\,2} = \frac{-\lambda_3 \cdot \lambda_1}{\lambda_2^2} \tag{9}$$

The distance between the camera and the target plane, denoted by *d*, can be obtained by applying triangle similarity principles to the triangle couples OO_cA-Oo_cx_0 and OO_cB-Oo_cr_0 shown in Fig. 6. From triangles OO_cA and Oo_cx_0 it can be inferred:

$$\frac{d}{1} = \frac{\delta}{x_0} \tag{10}$$

From triangles OO_cB and Oo_cr_0, and applying Eqs (9) and (10), an expression for δ in terms of the eigenvalues of C_n and the known radius of the outermost border of a target's bull's-eye, ρ, can be derived (see Eq. (11)). Similarly, substituting Eq (11) in Eq. (10), an expression for the distance of the target plane to the camera origin, *d*, is obtained (see Eq. (12)):

$$\frac{\rho}{r_0} = \frac{\delta}{x_0} \Rightarrow \delta = \sqrt{\frac{-(\lambda_2 - \lambda_1)(\lambda_3 - \lambda_2)}{\lambda_1 \cdot \lambda_3}} \cdot \rho \tag{11}$$

$$d = \frac{\rho}{r_0} = \sqrt{\frac{-\lambda_2^2}{\lambda_1 \cdot \lambda_3}} \cdot \rho \tag{12}$$

The 3D coordinates of the centre of the target (see Fig. 4) are given by the translation vector, $\vec{T}$, expressed in the OX''Y''Z'' frame. $\vec{T}$ can be calculated in terms of the original coordinate system *OXYZ* as:

$$\vec{T} = R_C \cdot \begin{bmatrix} \delta & 0 & d \end{bmatrix}^T \tag{13}$$

Again, as a consequence of Eq. (6), there are two possible solutions for the translation vector, *i.e.* $\vec{T}_1$ and $\vec{T}_2$.

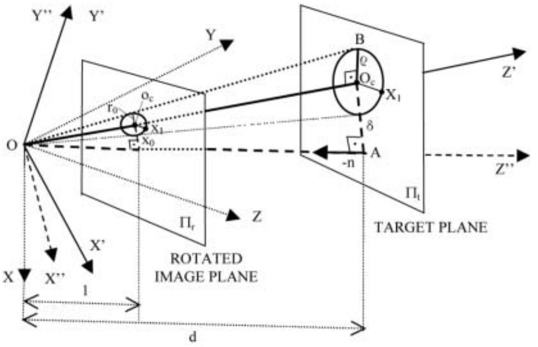


Fig. 6. Geometric relations between planes Π_r and Π_t.

2.3.3. Breaking the ambiguity

As noted by the previous two sections, the `POSE_FROM_TRIPTAG` method returns two possible pose solutions. This result can be explained geometrically by observing Fig. 7. Circle 1 and circle 2, lying on planes Π_1 and Π_2, respectively, and with the same radius r are projected to the same ellipse in the image plane. Consequently, it is necessary to use additional information, apart from the ellipse of reference, in order to be able of deciding which of the two solutions is the real one. For TRIP, this additional information is provided by the known projection of the bottom outermost corner of the synchronisation sector of a TRIPtag (see Fig. 1). The projection of this point is accurately identified by the code deciphering stage of the target recognition process, described in Section 2.2. Likewise, the location of this point in the target centred coordinate system is also known (see Fig. 3). If the projections of this point, obtained by means of the two rigid body transformations returned by the `POSE_FROM_TRIPTAG` method, are compared with the known projection of this point in the image, the right solution can be determined. The point projection lying closest to the known projection in the image will correspond to the right solution. Figure 8 shows how the projection of a point at a distance r+d from the centre of a circle in plane Π_1 will differ significantly from the projection of the same point lying on plane Π_2.

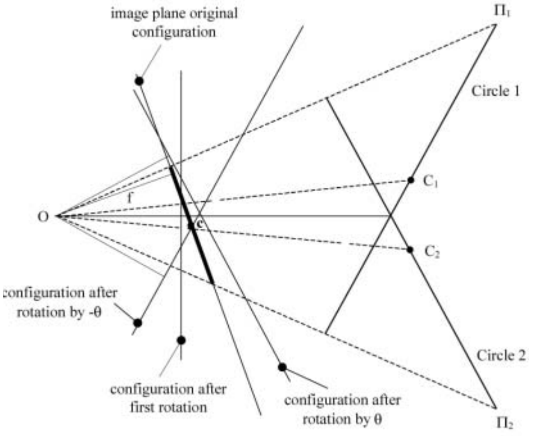


image plane original configuration
Π_1
Circle 1
C_1
O
f
c
C_2
configuration after rotation by -θ
configuration after first rotation
configuration after rotation by θ
Circle 2
Π_2

Fig. 7. Circle projection ambiguity.

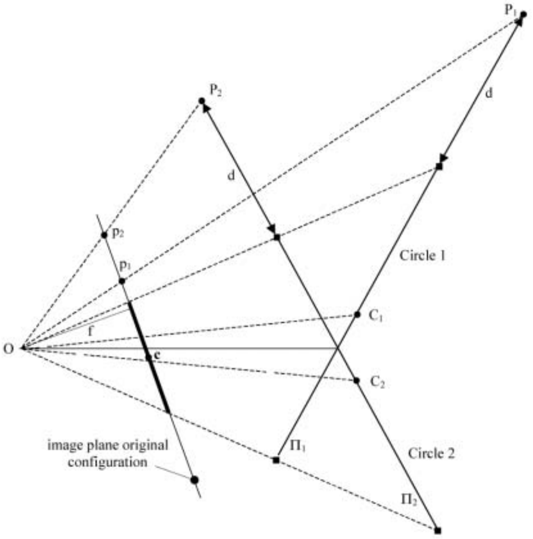


Fig. 8. Projections of points P_1 and P_2 at r+d distance from C_1 and C_2.

The projection of point $P=[2.2{\cdot}r,\ 0,\ 0,\ 1]^T$ (expressed in homogenous coordinates) in the TRIPtag plane into point $p=[su,\ sv,\ s]^T$ in the image plane can be obtained by applying the *planar projective transformation* [17]:

$$\begin{bmatrix} su \\ sv \\ s \end{bmatrix} = K \cdot \begin{bmatrix} R_{C_{11}} & R_{C_{12}} & T_x \\ R_{C_{21}} & R_{C_{22}} & T_y \\ R_{C_{31}} & R_{C_{32}} & T_z \end{bmatrix} \cdot \begin{bmatrix} 2.2 \cdot r \\ 0 \\ 1 \end{bmatrix} \tag{14}$$

The planar projective transformation exploits the fact that every point lying on the target plane has a value of Z=0. Equation (14) can be applied to the two feasible results obtained, i.e. (R_{C1} and $\vec{T}_1$) and (R_{C2} and $\vec{T}_2$), so as to calculate the projections p_1 and p_2. In order to measure the distances between p_1 and p (the known projection of the point in the image), *and* p_2 and p, the homogeneous coordinates of these points are transformed into Cartesian coordinates. The right solution should project P into a point lying very close to p:

$$\vec{T}, n = \left\{ \begin{matrix} \vec{T}_1, n_1 & if & \|p_1 - p\| & < & \|p_2 - p\| \\ \\ \vec{T}_2, n_2 & if & \|p_1 - p\| & \geqslant & \|p_2 - p\| \end{matrix} \right\} \quad (15)$$

2.3.4. Target's Rotation Angles

The use of a circle as a reference has provided a closed form solution to the determination of the 3D pose of a TRIPtag. Unfortunately, the view of a planar circle, due to its symmetry, does not permit the determination of rotations around the Z-axis of the target coordinate system, orthogonal to the target's plane. No matter how much a circle is rotated around the Z-axis, its projected image looks the same. This signifies that from the rotation matrix R_C, only the angles around the axes X and Y, α and β respectively, can be retrieved. Therefore, it is necessary to calculate a new rotation matrix, named $R_C{}'$ from which the angle γ around the Z axis can also be recovered.

The vector columns of the $R_C{}'$ matrix are, in fact, the three unitary vectors defining the TRIP target-centred coordinate system. From these vectors, the one corresponding to the target's Z-axis is already known and given by $\vec{n}$. In order to calculate the other two column vectors, namely $\vec{r}_x$ and $\vec{r}_y$, is necessary to use two correspondences between points expressed in the target's frame and their projections in the image plane. The first point correspondence is given by the known centre of the target, $\vec{T}$, and its projection in the image. The second correspondence is given by the back-projection of the bottom outermost corner of the synchronisation sector projection, denoted by a small triangle in Fig. 6 and named x_1 in Fig. 6, into X_1. This point can be uniquely identified in every projection of a TRIPtag. The following is applied to calculate this correspondence.

Given a vector $\vec{p} = [p_x \quad p_y \quad p_z]^T$ any other vector with the same direction and origin is given in homogenous coordinates by Eq. (16), where s indicates a free scale factor applied to the vector's modulus:

$$\overrightarrow{P(s)} = \begin{bmatrix} \vec{p} \\ s \end{bmatrix} \quad (16)$$

Therefore, the correspondence between the 3D point X_1, and its projection in the image x_1 is given by:

$$\overrightarrow{X_1} = \begin{bmatrix} \overrightarrow{Ox_1} \\ s_c \end{bmatrix} = \begin{bmatrix} x_{1x} \\ x_{1y} \\ x_{1z} \equiv 1 \\ s_c \end{bmatrix} \quad (17)$$

The scale factor s_c can be calculated by considering that the point X_1 belongs to the target plane Π_t, i.e.

$$\overrightarrow{P(s)}^T \cdot \Pi_t = 0 \quad (18)$$

where Π_t is given by the *point normal plane* equation:

$$\Pi_t = [x \quad y \quad z \quad 1] \cdot \begin{bmatrix} \vec{n} \\ -d \end{bmatrix} = 0 \quad (19)$$

Thus, substituting Eq. (19) in Eq. (18), and using the known coordinates of the bottom outermost corner of the synchronisation sector of the TRIPtag (x_{1x}, x_{1y}), the value of s_c is:

$$s_c = \frac{\overrightarrow{x_1}^T \cdot \vec{n}}{d} \quad (20)$$

Finally, the coordinates of X_1, expressed in the original coordinate system, $OXYZ$, is calculated by combining Eqs (20) and (17):

$$\overrightarrow{X_1} = \begin{bmatrix} x_{1x} / s_c \\ x_{1y} / s_c \\ (x_{1z} \equiv 1) / s_c \end{bmatrix} \quad (21)$$

$\vec{r}_x$ is given by the unitary vector in the direction $\vec{T} - \vec{X}_1$. To calculate $\vec{r}_y$, the cross product of $\vec{r}_x$ and $\vec{n}$ is performed. From the R_C' matrix the angle γ can now be recovered.

2.4. TRIP Sensor Operation Modes

The main limitation of vision-based sensors is their high processing demands. TRIP, despite being optimised to require the least possible processing load, is not an exception and still presents relatively high processing demands. Section 2.5 shows that a PC with a processor of 1.4 GHz manages to apply the TRIP processing to 16 image frames per second. Nevertheless, a TRIP sensor will often be analysing frames where no TRIPtags are encountered, or alternatively, frames where the approximate location of a TRIPtag in the image could be inferred based on its location in previous frames. Therefore, it is convenient to provide the TRIP sensor with some intelligence, both to distinguish the worthiness of analysing an image, and

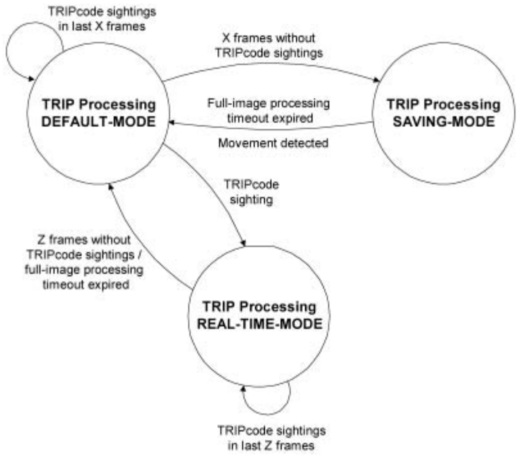

TRIPcode sightings in last X frames
X frames without TRIPcode sightings
TRIP Processing DEFAULT-MODE
Full-image processing timeout expired
TRIP Processing SAVING-MODE
Movement detected
TRIPcode sighting
Z frames without TRIPcode sightings / full-image processing timeout expired
TRIP Processing REAL-TIME-MODE
TRIPcode sightings in last Z frames

Fig. 9. Operating modes of a TRIP sensor.

if the analysis proceeds, to determine whether the full-image parsing should take place or this process should just be applied to a smaller subwindow in the image. The aim is to give to TRIP maximum responsiveness (real-time video parsing) but still guaranteeing the minimum consumption of processing load. For this reason, TRIP is provided with an *adaptive* behaviour during its operation. The sensor transitions through three different operating modes, named *default-mode*, *saving-mode* and *real-time-mode*, depending on the characteristics of the scene viewed. Figure 9 shows a state diagram with the different operation modes of the TRIP sensor and the conditions upon which the sensor switches from one to another.

The TRIP default-mode of operation is activated either on sensor start-up or whenever a timeout expires. The timeout, specified as a parameter in the system bootstrap, limits the time the sensor spends in either the saving or real-time modes, where full-image analysis does not take place. This is necessary to ensure a good responsiveness of the system to scene changes.

The TRIP saving-mode is triggered by the default-mode whenever there are no TRIPtag sightings in a given number of frames specified at the system start-up. This mode undertakes a *Triggering Analysis* by which TRIP decides whether it should return to the default mode (if significant changes in the scene have occurred) or otherwise stop consuming processing resources for a while, i.e. sleep. Basically, an efficient movement detection mechanism is implemented. Each time slot the triggering module analyses a low-resolution version of a newly captured image (i.e. one in every 5 pixels in both horizontal and vertical directions) and compares each pixel with a previously calculated *Running Video Average* (RVA) frame of the background. An RVA [21] is a method used to construct an evolving background frame, insensitive to small variations in illumination or the slight flicker of electric light, using the average pixel values of the N preceding images. The triggering stage determines the percentage P of pixels that have changed by more than a certain threshold. A high percentage means that movement has been detected and the parsing control is then passed to the default-mode. Otherwise the TRIP sensor stops its image processing (sleeps) for a short period and then the same triggering analysis process is repeated. When the full image processing timeout expires, control is passed back to the default mode.

The TRIP real-time-mode of operation is triggered by the default-mode each time a TRIPcode is spotted in an image. This mode exploits the spatial locality of TRIPtag images in successive video frames. Once a target has been identified within an image, its outermost elliptical border is tracked in subsequent frames. An ellipse tracking method undertakes edge detection along the normal to the tangents of the previous ellipse at various points, in order to identify the new location of points belonging to the new ellipse. Then, the method reapplies the ellipse-fitting algorithm with the newly found ellipse points. Finally, the `POSE_FROM_TRIPTAG` method is applied to the newly obtained ellipse parameters. In order to guarantee the rapid identification of newly appearing TRIPtags, when the full image processing timeout expires, control is passed back to the default mode. Furthermore, if after Z frames no TRIP sightings are found then control is also returned to the default-mode.

2.5. TRIP Performance and Accuracy

The current C++ implementation of the TRIP target recognition algorithm processes 17 640 × 480 pixel frames per second on a 1.4 GHz AMD Athlon processor. When the target recognition and pose estimation are simultaneously undertaken, the performance achieved on the same machine is about 16 Hz. Fig. 10 shows the time and percentage of the total image parsing time invested in each of the stages of the TRIP parsing process. Real-time video processing is

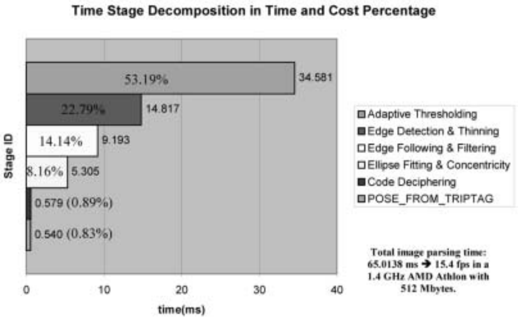


Fig. 10. TRIP sensor performance.

achieved through the adaptive operation behaviour illustrated in the previous section. A current off-the-shelf PC with a CPU over 1.5 GHz may be used to process images from three cameras simultaneously, still providing an average throughput higher than 6Hz. The processing cost of TRIP is not affected by the number of targets spotted within the image, since only the code deciphering and `POSE_FROM_TRIPTAG` stages require additional processing when many targets are seen. The processing cost of these two stages is really small compared to the others.

TRIPtags are recognised as long as the slant between the normal to the target plane and the translation vector $\vec{T}$ is less than 70° and the target image occupies an area of at least 35 × 35 pixels. Upon those conditions, recognition is successful on a 98% of the cases. False negatives are more common once these threshold limits are overtaken. So, targets spotted in a frame of 640 × 480 pixel resolution are identified as long as they are within 3 meters distance. The pose extraction method returns the 3-D location of the centre of a TRIP target with respect to a viewing camera with an average error of less than 3%. Figure 11 shows the error in cms during 3-D location estimation incurred by the TRIP system. The slant detected by TRIP is within a 2% error of the actual slant. The likelihood of finding false positives is negligible thanks to the error parity check used.

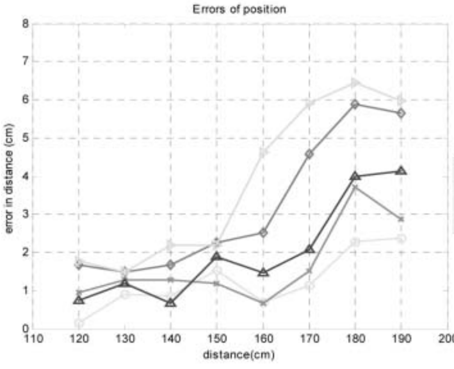


Fig. 11. TRIP sensor 3D location accuracy.

3. TRIP: A Distributed Sensor

An event-based distributed architecture has been devised around TRIP to manipulate and distribute the sensor data provided by this technology. The functionality of the TRIP system, *i.e.* its target recognition and pose extraction, has been encapsulated within a CORBA [22] component named *TRIP$_{parser}$*. This component, implemented using the omniORB CORBA ORB [23], offers a UNIX pipe-like interface which enables applications to connect distributed *Frame Grabber* components, obtaining images from cameras spread throughout the environment, to *TRIP$_{parsers}$*. A *TRIP$_{parser}$* may consume images supplied by one or more Frame Grabbers. Every frame submitted for processing is tagged with a camera identifier. TRIP processing results are, by default, asynchronously communicated in event-form to a CORBA Notification Service's Channel [24] associated with each *TRIP$_{parser}$*. This interleaved event communication component decouples analysers' frame processing and result reporting duties. Hence, TRIP can concentrate on the CPU demanding image parsing, whereas the Notification Channel deals with supplier and consumer registration, timely and reliable event delivery to registered consumers, and the handling of errors associated with unresponsive consumers.

A *TRIP$_{parser}$* generates *TRIP$_{events}$* (see Fig. 12) which are mapped into the *Structured Event* message type supported by the CORBA Notification Service [24]. Structured Events define a standard format for messages conveyed to a Notification Channel. In the body of this message the contents of an event are mapped into a set of name-value pairs to which filtering operations can be applied. Parties interested in a given *TRIP$_{parser}$*'s raw location data subscribe to its channel passing a set of constraints over those name-value pairs of an event, expressed in the Extended Trader Constraint Language [25]. The Notification Channel performs the event filtering and communication on behalf of its representing *TRIP$_{parser}$*. Hierarchical interconnections of *TRIP$_{parsers}$*' Notification Channels can be

```
struct TRIPevent {
    TimeStamp time; // frame captured timestamp
    string TRIPcode; // code ternary representation
    string cameraID; // capturing camera identifier
    paramsEllipse params;
    // bull's-eye's outer ellipse params(x,y,a,b, )
    targetPosition pose; // (xpos, ypos, zpos)
    targetOrientation angles; // (α, β, γ)
    };
```

Fig. 12. TRIPevent contents.

created in order to ensure the efficient and scalable dissemination of TRIP sightings. For example, $TRIP_{parsers}$ running in hosts within a room could push their events to the same event channel, and similar notification channels for other rooms could be federated, in order to make available the whereabouts of TRIPtag wearers within a building.

A $TRIP_{parser}$ also provides a synchronous invocation interface (`processFrame`) that analyses the frame passed as the argument and returns the location data inferred from it. Hence, applications can interact with a $TRIP_{parser}$ in either a synchronous or asynchronous form. However, for efficiency purposes, it is encouraged that applications establish direct communication pipes between Frame Grabbers and $TRIP_{parsers}$, and register as event consumers of the parsers' Notification Channels.

3.1. The TRIP Directory Service

A TRIP Directory Server (TDS) has been created with the purpose of regulating the TRIPcode granting process and establishing mappings between real-world objects and TRIPcodes. This component ensures the efficient utilisation of the TRIPcode address space and its classification into categories with a common ternary prefix, used by consumers in event filter registration. The TDS offers CORBA interfaces for the creation, modification, deletion and retrieval of both TRIPcodes and their categories.

4. TRIP-enabled Applications

TRIP has been employed in the development of several context-aware applications. In this section, four significant examples of its usability in both off-line and real-time video parsing are given.

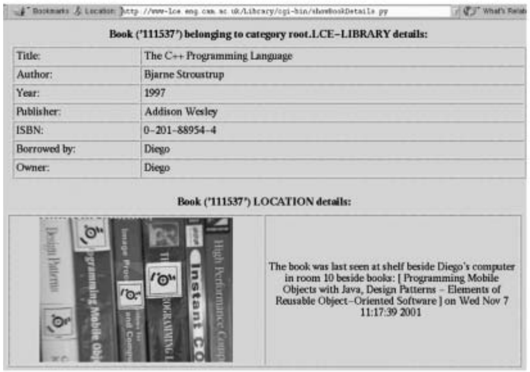
4.1. LCE Sentient Library

This system augments a conventional library catalogue system with sentient features (see Fig. 13). Apart from the typical functionalities expected in a book cataloguing system, this one offers contextual information about the books in the LCE laboratory. Details on the last seen book location and its proximity to other fixed items in the environment are offered.

Every location where a book may be located in the LCE has been tagged with a *location-category* TRIPcode. Similarly, *book-category* TRIPcodes have been attached to book spines. Periodically, the LCE librarian wanders around our lab with a video camera recording TRIP tagged locations and books. The system automatically updates the book database by the off-line processing of the booksightings video. This process involves the co-operation of a Video Frame Grabber, which provides access to the video frames, a $TRIP_{parser}$, which analyses those frames, and the TRIP Directory Server, where book details and contextual data are recorded.

A web interface allows LCE members to (1) browse through book categories and the books in a category, (2) perform keyword-based search of books, (3) create new book categories, (4) input new books' details and printout their associated TRIPtag and (5) modify book details. Figure 9 shows the result of a book search with this web interface.

This application is an illustration of TRIP's versatility for tracking any tag-able entities in the environment. TRIP's costless resizable tags make this positioning technology suitable even for the location of inexpensive items in the physical space.


Bookmarks
Location: http://www-lce.eng.cam.ac.uk/Library/cgi-bin/showBookDetails.py
What's Related

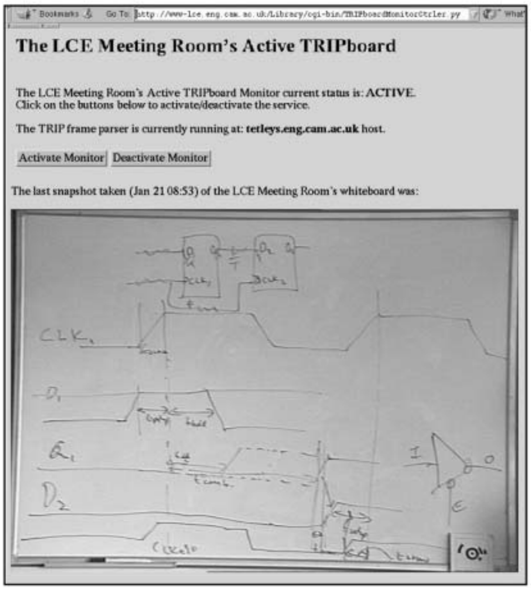
Book ('111537') belonging to category root.LCE–LIBRARY details:

Title:	The C++ Programming Language
Author:	Bjarne Stroustrup
Year:	1997
Publisher:	Addison Wesley
ISBN:	0–201–88954–4
Borrowed by:	Diego
Owner:	Diego

Book ('111537') LOCATION details:

The book was last seen at shelf beside Diego's computer in room 10 beside books: [Programming Mobile Objects with Java, Design Patterns – Elements of Reusable Object–Oriented Software] on Wed Nov 7 11:17:39 2001

Fig. 13. LCE Sentient Library Web Interface.



Bookmarks
Go To: http://www-lce.eng.cam.ac.uk/Library/cgi-bin/TRIPboardMonitorCtrler.py
What's
Activate Monitor
Deactivate Monitor
CLK1
D1
Q1
D2
I
O
E

The LCE Meeting Room's Active TRIPboard

The LCE Meeting Room's Active TRIPboard Monitor current status is: **ACTIVE**.
Click on the buttons below to activate/deactivate the service.

The TRIP frame parser is currently running at: **tetleys.eng.cam.ac.uk** host.

The last snapshot taken (Jan 21 08:53) of the LCE Meeting Room's whiteboard was:

Fig. 14. Active TRIPboard snapshot.

4.2. Active TRIPboard

This application augments a conventional whiteboard placed in a meeting room with interactive commands issued by placing TRIPtags in the field of view of a camera focused on the whiteboard. Some example actions that can be triggered are: (1) capture whiteboard snapshot and email its web link to people currently located in the room, (2) printout a copy of the whiteboard snapshot for each person in the room. The application components: a Frame Grabber and a *TRIP*$_{parser}$ are activated, via the LocALE [26] CORBA object activation framework, either when a person appears in the meeting room or through a web interface (see Fig. 14). The Active Badge indoor location system is used to determine person presence. In future, the meeting room will be populated with enough cameras to cover the entire space, and so permit people wearing TRIPtags to be detected. This application is an illustration of how TRIP can be used as a device for human-computer interaction. Moreover, it combines two sensor outputs, Active Badge and TRIP, in order to enact actions for the user.

4.3. Follow-Me Audio

This application provides mobile users with music from the nearest speakers wherever they are. The source of music is an MP3 jukebox server whose operation is controlled by showing jukebox-category TRIPcodes to web-cams attached to some of our lab's PCs. A personal software agent, associated with each lab person, listens for that TRIP wearer's movement events generated by a *Person Location Monitor*. This component obtains people sightings from the Notification Channels of the *TRIP*$_{parsers}$ corresponding to the PC web-cams spread throughout the laboratory. From raw TRIP sighting events, it generates personnel presence and movement events. Similarly, the personal agent registers with a *Jukebox Controller Context Abstractor* that maps TRIP sightings into jukebox control events. The personal agent, acting as a component migration controller, requests the LocALE [26] object lifecycle and location control middleware to migrate the audio player and MP3 decoder components to the host nearest to the user that can deliver the music. As the user wanders around the location-aware computing environment, the music follows him. The state of the system and time index into the current song persists as the components migrate. Likewise, the agent issues control operations to the jukebox server when jukebox-control events are received. This application shows TRIP's capabilities as a real-time identification and location sensor.

4.4. TRIP Teleporting

The TRIP-enabled Teleporting service monitors when a TRIPtag-wearing user gets closer than one metre to any of the web-cams placed on top of the computers in the LCE laboratory, and those machines are not currently being used. When this situation is matched, the service automatically displays the desktop associated with that user through Virtual Network Computing (VNC) [27], as shown in Fig. 15. When the user later moves in front of another terminal, the VNC desktop is automatically deactivated from the previous terminal and activated on the new one. This application shows TRIP's capability as a fine-grained location sensor, capable of measuring the precise location of objects in space.

5. Related Work

SONY's CyberCode [14] is a visual tagging system based on a 2-D square barcode that alike to TRIP can be used to determine the 3-D

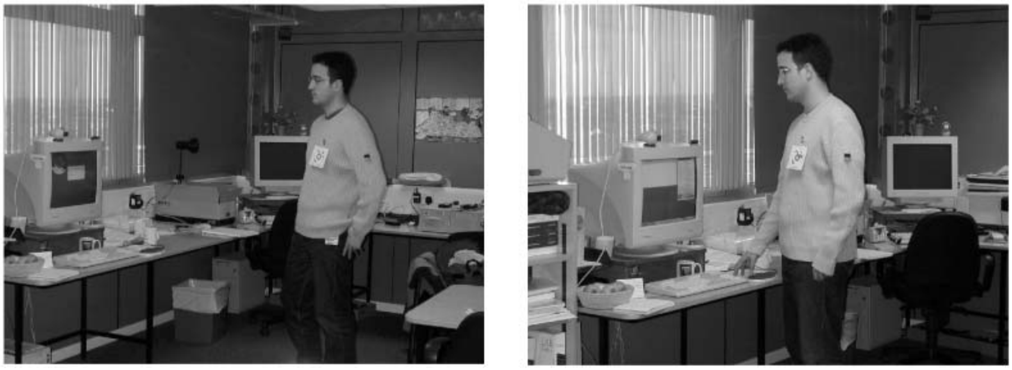


Fig. 15. VNC teleport when user approaches to host.

position and identifier of tagged objects. The ARToolKit [2] system also chooses a square marker for a similar purpose, however, its identifier encoding capabilities are more limited than in the case of TRIP or CyberCode. Both CyberCode and ARToolKit technologies provide a high degree of location accuracy and work in realtime. They have been mainly applied to the domain of Augmented Reality, in order to correctly register computer-synthesised information on views of the real world. However, these square marker technologies are not as easily identifiable as TRIP circular targets in cluttered environments and therefore are less suitable for the object location and tracking domain we target. Moreover, their marker geometric features require higher image resolution for accurate recognition and location extraction than TRIP.

BBC's free-d [12] location system measures the precise position and orientation of studio cameras, by using an auxiliary camera mounted on the back of a conventional moving camera pointing to circular markers, similar to TRIPcodes, fixed on the ceiling of a TV recording studio. A hardware implementation of its algorithms is needed to achieve real-time video processing. The system is used for virtual reality TV production, being, in contrast to TRIP, expensive and cumbersome to deploy. Xerox PARC's Dataglyph [28] is a 2D barcode technology that uses thousands of tiny, individual 45-degree diagonal lines to encode information. This barcode offers a high density encoding and is robust to partial occlusion. A derivation of Dataglyph, requiring less resolution for its identification, has been used to enhance a conventional notice board with electronic services, alike to the TRIPboard application. Nevertheless, Dataglyph requires far higher resolution for the tag identification and location than TRIP.

6. Further Work and Conclusions

So far, mainly the entity presence identification capability of TRIP has been exploited. Current ongoing work is addressing the creation of sentient applications that make use of the precise 3-D location and orientation information also provided by TRIP. An example application in mind is the use of a TRIPtag as a 3-D mouse for the control of virtual 3-D environments, e.g. in the Quake game. We also hope to deploy, in due time, sufficient number of web-cams across our laboratory in order to make possible the creation with TRIP of a 3D Location Service covering all our premises.

TRIP is a novel cost-effective and easily deployable sensor technology that offers an excellent degree of accuracy and performance for the identification and 3D location of tagged entities. This sensor's off-the-shelf hardware requirements, i.e. inexpensive low-resolution CCD cameras, monochromatic printers to generate its 2D circular tags and PCs, makes the creation of location-aware reactive environments, even in the home, an affordable proposition. All that is required to augment a standard PC with visual awareness is a standard web-cam which costs less than $50 and TRIP's software. It is hoped that our system will assist in a wider acceptance of the promising Sentient Computing paradigm which provides a more natural and flexible way of interfacing with computers. The set of TRIP-enabled applications presented demonstrates some of the potential of this low-cost but high performing 3-D location system.

Acknowledgements

Diego López de Ipiña would like to express his gratitude to the Department of Education of the Basque Government for the sponsorship of his PhD studies. The TRIP project has been sponsored by AT&T Laboratories, Cambridge.

References

1. Weiser M. The computer for the 21st Century. Scientific American 1992; 265(3): 94–104
2. Hopper A. Sentient computing. Philosophical Transactions of the Royal Society London 2000; 358 (1773): 2379–2358
3. Addlesee M, Curwen R, Hodges S, Newman J, Steggles P, Ward A, Hopper A. Implementing a sentient computing system. IEEE Computer 2001; 34(8): 50–56
4. Brumitt B, Meyers B, Krumm J, Kern A, Shafer S. EasyLiving: Technologies for intelligent environments. Handheld and Ubiquitous Computing September 2000
5. Want R, Hopper A, Falcão A, Gibbons J. The Active Badge location system. ACM Transactions on Information Systems 1992; 10(1): 91–102
6. Harter A, Hopper A, Steggles P, Ward A, Webster P. The anatomy of a context-aware application. Proceedings of MOBICOM'99, Seattle, WA 1999
7. Ward A, Jones A, Hopper A. A new location technique for the active office. IEEE Personal Communications 1997; 42–47
8. Werb J, Lanzl C. Designing a positioning system for finding things and people indoors. IEEE Spectrum 1998; 71–78
9. Hightower J, Borriello G. Location systems for ubiquitous computing. IEEE Computer 2001; 34(8): 57–66
10. Krumm J, Harris S, Meyers B, Brumitt B, Hale M, Shafer S. Multi-camera multi-person tracking for EasyLiving. Proceedings 3rd International Conference on Visual Surveillance, Dublin, Ireland, July 2000; 3–10
11. Stillman S, Tanawongsuwan R, Essa I. A system for tracking and recognizing multiple people with multiple cameras. Proceedings 2nd International Conference on Audio-Vision-based Person Authentication, Washington, DC April 1999
12. Thomas GA, Jin J, Niblett T, Urquhart C. A versatile camera position measurement system for virtual reality in TV production. Proceedings IBC'97 September 1997; 284–289
13. Kato H, Billinghurst M. Marker tracking and HMD calibration for a video-based augmented reality conferencing system. Proceedings 2nd Internation Workshop on Augmented Reality 1999; 85–94
14. Rekimoto J, Ayatsuka Y. CyberCode: designing augmented reality environments with visual tags. Proceedings of DARE 2000
15. Pilu M, Fitzgibbon A, Fisher R. Ellipse-specific Direct least-square Fitting. IEEE International Conference on Image Processing September 1996
16. Wellner P. Interacting with paper on the DigitalDesk. Communications of the ACM 1993; 36(7): 87–96
17. Trucco E, Verri A. Introductory Techniques for 3D Computer Vision. Prentice-Hall 1998
18. Forsyth D, Mundy JL, Zisserman A, Coelho C, Heller A, Rothwell C. Invariant descriptors for 3-D object recognition and pose. IEEE Transactions on Pattern Analysis and Machine Intelligence 1991; 13(10): 971–991
19. Tsai RY. A versatile camera calibration technique for high-accuracy 3-D machine vision metrology using off-the-shelf TV cameras and lenses. IEEE Journal of Robotics and Automation, August 1987; RA-3(4): 323–344
20. Zhang Z. A flexible new technique for camera calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence 2000; 22(11): 13301334 (software available at: http://research.microsoft.com/~zhang/Calib/)
21. Stafford-Fraser JQ, Robinson P. BrightBoard: a videoaugmented environment. Proceedings Conference on Human Factors in Computing Systems (CHI96) April 1996
22. Object Management Group. The Common Object Request Broker Architecture: Architecture and Specification, October 1999 (ftp://ftp.omg.org/pub/ docs/formal/01-09-01.pdf)
23. OmniORB 3.0 High Performance ORB, AT&T Laboratories Cambridge, 2001 (http://www.uk.research.att.com/omniORB/index.html)
24. Object Management Group. Notification Service, November 1998 (ftp://ftp.omg.org/pub/docs/formal/00-06-20.pdf)
25. Object Management Group. Trading Object Service Specification, May 2000 (ftp://ftp.omg.org/pub/docs/formal/00-06-27.pdf)
26. López de Ipiña D, Lo S. LocALE: a location-aware lifecycle environment for ubiquitous computing**.** Proceedings 15th International Conference on Information Networking (ICOIN-15) February 2001
27. Richardson T, Stafford-Fraser Q, Wood KR, Hopper A. Virtual network computing. IEEE Internet Computing 1998; 2(1): 33–38
28. Moran TP, Saund E, Van Melle W, Gujar AU, Fishkin KP, Harrison BL. Design and technology for Collaborage: collaborative collages of information on physical walls. Proceedings 12th Annual ACM Symposium on User Interface Software and Technology November 1999

Correspondence to: Diego López de Ipiña, 134b Cherry Hinton Road, Cambridge, CB1 7AJ. Email: d1231@eng.cam.ac.uk